

INVERSE SIMULATION AS A TOOL FOR FAULT DETECTION & ISOLATION IN PLANETARY ROVERS

**Murray L Ireland, Rebecca Mackenzie, Thaleia Flessa, Kevin Worrall, Douglas Thomson,
Euan McGookin**

*School of Engineering, University of Glasgow, Glasgow, UK, G12 8QQ, Contact:
Murray.Ireland@glasgow.ac.uk*

ABSTRACT

Fault detection, isolation and recovery is crucial in semi- and fully-autonomous vehicles, particularly such vehicles as planetary rovers, which are far-removed from assistance, human or otherwise, in the event of a fault. Residual generation is a popular and conceptually simple method of model-based fault detection, comparing estimates of system properties with other estimates or measurements of those properties. Output residual generation is a simple example of this, employing a mathematical model of the nominal system behaviour, which is driven by the measured system inputs. It then produces an estimate of the system output, which is compared to the output of the real, fault-afflicted system in the form of an output residual error. In this paper, the generation of residuals for variables beyond the output is considered. Where output residual generation employs a standard model of the system, other residuals require inverse models of the system or its subsystems. This inversion is achieved using Inverse Simulation (InvSim). The use of InvSim is shown to enable generation of residuals at the input and between subsystems. These residuals demonstrate greater clarity in certain faults than output residuals. Additionally, InvSim can produce different results when driven by different subsets of the output. This functionality permits discussion of an architecture which can employ forward simulation and multiple InvSim modules to generate a large suite of residuals for fault detection and isolation.

1 INTRODUCTION

Robotic rovers are at the forefront of humanity's exploration of Mars. Vehicles such as Sojourner, Spirit, Opportunity and Curiosity have provided invaluable data on the red planet, deepening our understanding of it and paving the way for future manned missions. The success of these missions is strongly dependent on the robustness of the rover and its ability to adapt to problems raised by the environment and its own systems. Millions of kilometres from any direct assistance and operating in an unpredictable, hostile environment, a rover can have its mission compromised by any number of events. A fault in the rover's systems, for example, can result in the rover operating incorrectly or, at worst, becoming completely immobile. For every Spirit and Opportunity that far outlive their mission life, there is a Yutu that becomes immobile a fraction of the way through its mission, as a consequence of suffering a fault [1]. The need for robust and intelligent fault tolerance in such vehicles is clear.

Fault detection, isolation and recovery (FDIR) concerns the detection, localisation and attenuation of faults in a system. With the increasing complexity in systems and the increasing autonomy in

their control, FDIR has become a topic of great importance. FDIR techniques are many; one of the most conceptually simple is that of model-based detection [2]. This involves the use of a model of the nominal system behaviour to predict system properties. These predicted properties may be compared to those of the actual system. The comparison results in the generation of features, which ultimately allow any faults in the system to be isolated and attenuated. Residuals are one such feature, describing the difference between a system parameter and its nominal value. Output residuals are a common residual type, owing to their trivial estimation through a system model. Generation of residuals elsewhere in the system is less trivial, due to the lack of measurements or difficulty in obtaining estimates in these locations. In this paper, the generation of residuals in several locations of the rover system is investigated. Inverse Simulation (InvSim) is used to numerically invert a model of the rover and provide more flexibility in the generation of residuals.

2 THEORETICAL BASIS

Residual generation relies on sufficient information on the system behaviours being available. This allows properties of the system to be estimated based on available data such as system inputs or outputs. A typical example of such an approach is the generation of output residuals, where the system output is compared with an estimated output. The estimated output is obtained by supplying a mathematical model of the fault-free system with the system input. The difference between the output and estimated output is then known as the output residual. This method is effective when a fault occurs at the output of the system. Consider a generic system, described in the Laplace domain by

$$\mathbf{y}(s) = \mathbf{G}(s)\mathbf{u}(s) \quad (1)$$

where \mathbf{u} is the input, \mathbf{y} is the output and \mathbf{G} describes the system process. The system may be subject to an additive fault at the output, \mathbf{f}_y . In this event, the output is given by

$$\mathbf{y}(s) = \mathbf{G}(s)\mathbf{u}(s) + \mathbf{f}_y(s) \quad (2)$$

The output residual \mathbf{r}_y is generated as shown in Figure 1a, where the output $\hat{\mathbf{y}}$ of the fault-free model is subtracted from the fault-afflicted true output \mathbf{y} . The output residual is then

$$\begin{aligned} \mathbf{r}_y(s) &= \mathbf{y}(s) - \hat{\mathbf{y}}(s) = \mathbf{G}(s)\mathbf{u}(s) + \mathbf{f}_y(s) - \hat{\mathbf{G}}(s)\mathbf{u}(s) \\ &= \mathbf{f}_y(s) \end{aligned} \quad (3)$$

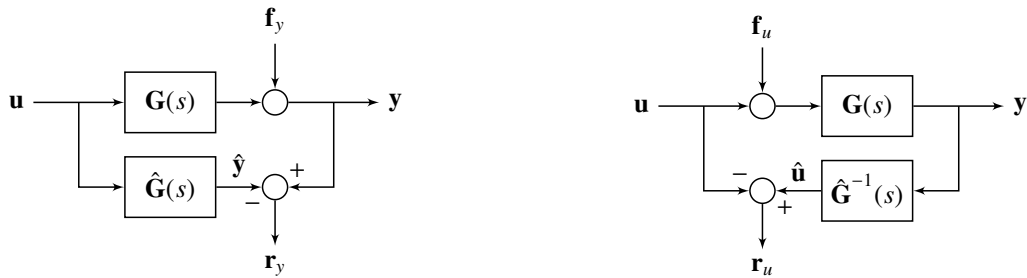
for the case where the process \mathbf{G} and process model $\hat{\mathbf{G}}$ match exactly.

The theoretical basis for input residuals may be constructed similarly [3]. Consider a system subject to a fault at the input and described in the Laplace domain by

$$\mathbf{y}(s) = \mathbf{G}(s) (\mathbf{u}(s) + \mathbf{f}_u(s)) \quad (4)$$

Equation (4) may be inverted such that an expression for input is obtained

$$\mathbf{u}(s) = \mathbf{G}^{-1}(s)\mathbf{y}(s) - \mathbf{f}_u(s) \quad (5)$$



(a) Process for generating output residuals.

(b) Process for generating input residuals.

Figure 1: Generation of output and input residuals using forward and inverse models.

where it is assumed that the system process \mathbf{G} is invertible. The input residual is then generated as shown in Figure 1b. As the fault term in Equation (5) is negative, the true input \mathbf{u} is subtracted from the modelled input $\hat{\mathbf{u}}$ to yield

$$\begin{aligned} \mathbf{r}_u(s) &= \hat{\mathbf{u}}(s) - \mathbf{u}(s) = \hat{\mathbf{G}}^{-1}(s)\mathbf{y}(s) - \mathbf{G}^{-1}(s)\mathbf{y}(s) + \mathbf{f}_u(s) \\ &= \mathbf{f}_u(s) \end{aligned} \quad (6)$$

where it is again assumed that $\mathbf{G} = \hat{\mathbf{G}}$. It is clear that output residual generation relies on a model of the system, while input residual generation relies on the inverse of this model. A combination of the two approaches may be employed to enable detection and isolation of faults in locations which are not immediately at the input and output. Consider some variable property \mathbf{p} of the system. The system is subject to a fault \mathbf{f}_p occurring at the location shown in Figure 2. The variable \mathbf{p} is then related to the system input and output by the expressions

$$\mathbf{y}(s) = \mathbf{A}(s)\mathbf{p}(s), \quad \mathbf{p}(s) = \mathbf{B}(s)\mathbf{u}(s) + \mathbf{f}_p(s) \quad (7)$$

Unlike the input and output, direct observations of \mathbf{p} are not available. Therefore, rather than comparing an estimated value with an observed one, two estimates of the same parameter are compared, one, $\hat{\mathbf{p}}_u$, derived from the system input \mathbf{u} and subsystem model $\hat{\mathbf{B}}$, and the second, $\hat{\mathbf{p}}_y$, derived from the system output \mathbf{y} and inverted subsystem model $\hat{\mathbf{A}}^{-1}$. The residual is then the difference between the two estimates

$$\begin{aligned} \mathbf{r}_p(s) &= \hat{\mathbf{p}}_y(s) - \hat{\mathbf{p}}_u(s) = \hat{\mathbf{A}}^{-1}(s)\mathbf{y}(s) - \hat{\mathbf{B}}(s)\mathbf{u}(s) = \hat{\mathbf{A}}^{-1}(s)\mathbf{A}(s)(\mathbf{B}(s)\mathbf{u}(s) + \mathbf{f}_p(s)) - \hat{\mathbf{B}}(s)\mathbf{u}(s) \\ &= \mathbf{f}_p(s) \end{aligned} \quad (8)$$

for the assumptions $\mathbf{A} = \hat{\mathbf{A}}$, $\mathbf{B} = \hat{\mathbf{B}}$.

This approach is analogous to the equation error residual described in [2], but is denoted *process residual generation* to relate it directly to the isolation of faults within the system process, as opposed to at the input and output. Both this method and input residual generation require that the relevant system, whether subsystem or full process, is invertible. For simple models, this inversion may be achieved symbolically. For a complex model such as the rover model presented in this paper, the inversion is achieved numerically using Inverse Simulation.

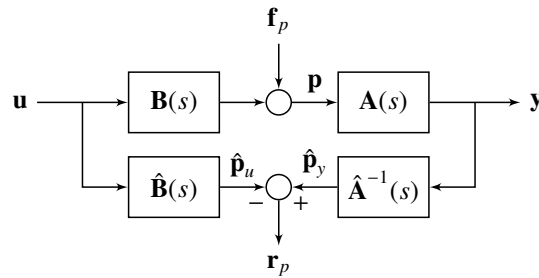


Figure 2: Generation of process residuals for detection and isolation of faults with system process.

3 INVERSE SIMULATION

Inverse Simulation (InvSim) is an iterative algorithm which numerically inverts a dynamic system. Where a conventional simulation employs a system model which receives an input and provides a corresponding output, InvSim does the reverse. For a given set of outputs, InvSim attempts to determine the inputs which will result in the system producing these outputs, within a specified tolerance. This technique is most often used in the context of identifying trajectories which a mechanical system can feasibly follow. Examples of this application include the helicopter [4, 5, 6, 7], autonomous underwater vehicle [8], unmanned aerial vehicle [9] and robotic rover [10, 11]. It differs from analytical approaches to system inversion such as non-linear dynamic inversion in that it may consider systems which contain discontinuities or are not *control-affine*. InvSim has also been used in model validation [6, 7].

InvSim uses a Newton-Raphson algorithm which is executed at discrete intervals during operation. It is initialised with approximate values for state and input, and then supplied with an output to track. This output may be either a desired trajectory or the true system output, but it must be sufficiently smooth with respect to the dynamics of the system. At each time step t_k , the Newton-Raphson algorithm attempts to converge on a solution for input $u(t_k)$, based on the supplied output $y(t_k)$ and state and input from the previous interval, $x(t_{k-1})$ and $u(t_{k-1})$, respectively. Performance of the algorithm may be improved by driving each step with a derivative of the output $y^{(a)}$ [6]. Unlike NDI, where the differential degree of the driving output is fixed by the equations of motion, the differential order of the driving output in InvSim can be varied. Its value does, however, impact the stability of the InvSim algorithm and must be chosen with care. Upon the Newton-Raphson algorithm converging on a solution, the state and input are recorded and used to initialise the algorithm at the next interval. There are two main implementations of InvSim: the differentiation and integration approaches. Of the latter approach, the most prominent solution is the *Genisa* algorithm [12], the behaviour of which is illustrated in Figure 3. An updated version of *Genisa* [10, 11, 13, 14] is employed in the investigation presented in this paper.

InvSim operates most effectively when the number of system inputs and outputs are equal [7], resulting in a square Jacobian J . It is therefore desirable to narrow the available system outputs to a few of interest. Where the driving outputs are desired trajectories, the outputs chosen are those which define the trajectory, such as position, velocity or heading. Where the driving outputs are the true system

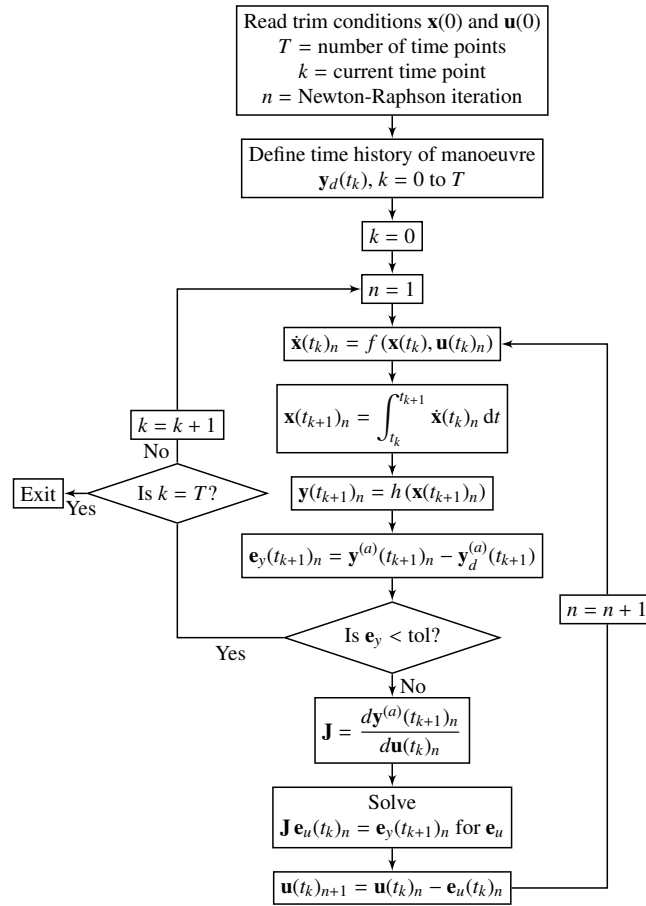


Figure 3: Genisa Inverse Simulation algorithm.

outputs, as is the case here, they must be chosen such that the effects of the fault are observable. When using signals obtained from the system itself, it is possible to use direct sensor outputs such as accelerometer, gyroscope and encoder measurements.

4 ROVER MODEL

The rover simulation and residual generation models employ a mathematical model, based on the Lynxmotion 4WD3 (Figure 4) and described comprehensively in [3, 15]. The model employs a 6 degree-of-freedom rigid body model dynamic motor model, however, the motion of the vehicle is limited to the horizontal plane for the purposes of this investigation. Model and controller properties are provided in [3].

4.1 Kinematics

The rover is described as a rigid body with position $\mathbf{r} = [x, y, z]^T$ and orientation $\boldsymbol{\eta} = [\phi, \theta, \psi]^T$ in an inertially-fixed reference frame \mathcal{W} . The velocities, forces and moments of the rover are described in a body-fixed reference frame \mathcal{B} , illustrated in Figure 5. The linear velocity $\mathbf{v} = [u, v, w]^T$ is related to

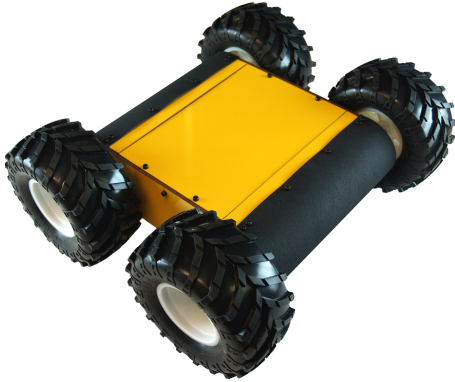


Figure 4: Lynxmotion 4WD3 rover.

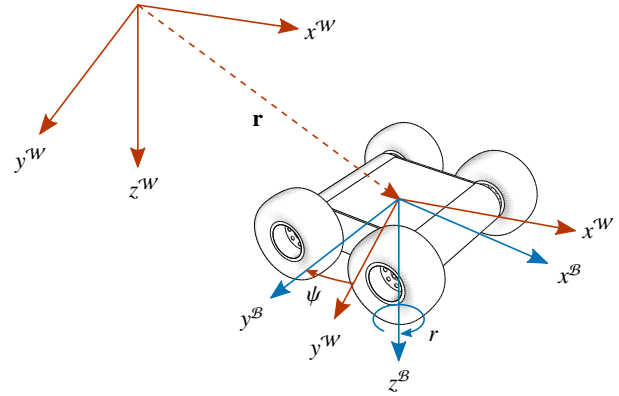


Figure 5: Rover frames of reference.

the inertial velocity by

$$\dot{\mathbf{r}} = \mathbf{R}_{\mathcal{B}}^{\mathcal{W}} \mathbf{v} \quad (9)$$

where the rotation matrix $\mathbf{R}_{\mathcal{B}}^{\mathcal{W}}$ is

$$\mathbf{R}_{\mathcal{B}}^{\mathcal{W}} = \begin{bmatrix} \cos \theta \cos \psi & \sin \phi \sin \theta \cos \psi - \cos \phi \sin \psi & \cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi \\ \cos \theta \sin \psi & \sin \phi \sin \theta \sin \psi + \cos \phi \cos \psi & \cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi \\ -\sin \theta & \sin \phi \cos \theta & \cos \phi \cos \theta \end{bmatrix} \quad (10)$$

and the reverse transformation is simply given by the transpose, $\mathbf{R}_{\mathcal{W}}^{\mathcal{B}} = (\mathbf{R}_{\mathcal{B}}^{\mathcal{W}})^T$. The angular velocity $\boldsymbol{\omega} = [p, q, r]^T$ is similarly related to the Euler rates by

$$\dot{\boldsymbol{\eta}} = \mathbf{R}_{\boldsymbol{\eta}} \boldsymbol{\omega}, \quad \text{where } \mathbf{R}_{\boldsymbol{\eta}} = \begin{bmatrix} 1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi \sec \theta & \cos \phi \sec \theta \end{bmatrix} \quad (11)$$

4.2 Rigid Body Dynamics

The rigid body dynamics describe the linear and angular velocities in response to a driving force \mathbf{F} and moment \mathbf{M} , that is

$$\dot{\mathbf{v}} = \frac{1}{m} \mathbf{F} - \boldsymbol{\omega} \times \mathbf{v}, \quad \dot{\boldsymbol{\omega}} = \mathbf{I}^{-1} (\mathbf{M} - \boldsymbol{\omega} \times \mathbf{I} \boldsymbol{\omega}) \quad (12)$$

where \mathbf{v} and $\boldsymbol{\omega}$ are the linear and angular velocities in \mathcal{B} , m is the rover mass and \mathbf{I} is the inertia tensor. The net force and moment have the contributions

$$\mathbf{F} = \underbrace{\sum_{j=1}^4 \frac{\tau_j}{R_w} \cdot \begin{bmatrix} \cos \beta \\ \sin \beta \\ 0 \end{bmatrix}}_{\text{Propulsive}} - \underbrace{\frac{1}{2} \rho C_d \|\mathbf{v}\|^2 \cdot \begin{bmatrix} A_x \cos \beta \\ A_y \sin \beta \\ 0 \end{bmatrix}}_{\text{Aerodynamic}} - \underbrace{mg \boldsymbol{\sigma}_v \mathbf{v}}_{\text{Frictional}} \quad (13)$$

$$\mathbf{M} = \underbrace{\sum_{j=1}^4 r_w \frac{\tau_j}{R_w} d_j \cdot \hat{\mathbf{z}}}_{\text{Propulsive}} - \underbrace{mgr_w \boldsymbol{\sigma}_\omega \boldsymbol{\omega}}_{\text{Frictional}} \quad (14)$$

where τ_j is the torque generated by wheel $j = \{1, 2, 3, 4\}$, d_j is the j th element of $\mathbf{d} = [1, 1, -1, -1]^T$, R_w is the wheel radius, r_w is the wheel moment arm, ρ is the atmospheric density, C_d is the body drag coefficient, A_x, A_y are body surface areas projected in $x^{\mathcal{B}}, y^{\mathcal{B}}$, respectively, g is the acceleration due to gravity and $\boldsymbol{\sigma}_v, \boldsymbol{\sigma}_\omega$ are the diagonal matrices of frictional coefficients for linear and angular motions, respectively. The slip angle β is determined by the direction of the velocity vector \mathbf{v} in the body frame, that is

$$\beta = \arcsin\left(\frac{v}{\|\mathbf{v}\|}\right) \quad (15)$$

4.3 Motor Dynamics

Each wheel $j = \{1, 2, 3, 4\}$ is driven by a DC motor which comprises an electrical component

$$\dot{i}_j = \frac{1}{L} (V_j - Ri_j - K_e \Omega_j) \quad (16)$$

and a mechanical component

$$\dot{\Omega}_j = \frac{1}{J_m} (K_t i_j - b \Omega_j - \xi \Omega_j) \quad (17)$$

where V is the voltage applied across the motor terminals, i is the current, L is the inductance, R is the resistance, K_e is the EMF constant, Ω is the motorspeed, J_m is the motor inertia, K_t is the torque constant, b is the viscous torque constant and ξ is the base friction coefficient. Motors on each side are paired such that $V_1 = V_2 = V_l$ and $V_3 = V_4 = V_r$, where V_l and V_r are the voltages supplied to the left- and right-hand motors, respectively. Each motor generates a torque τ which is related to the current and an efficiency factor η . The torque of a motor j is then

$$\tau_j = K_t i_j \eta_j, \quad \text{where } \eta_j = \alpha i_j + \gamma \quad (18)$$

where α, γ are constants determined through system identification. The motor torque drives the rover rigid body response through the propulsive force and moments, thus acting as the interface between the motor and rigid body subsystems.

4.4 Sensors

The sensors are modelled on an Optitrack motion capture system, a 6-axis inertial measurement unit (IMU) and four encoders for each of the four motors. The Optitrack system measures the three-dimensional position and orientation of the rover, respectively described by

$$\hat{\mathbf{o}}_{\text{pos}} = \mathbf{r}, \quad \hat{\mathbf{o}}_{\text{att}} = \boldsymbol{\eta} \quad (19)$$

The IMU includes triaxial accelerometers and gyroscopes. These measure specific force and angular rates and are respectively described by

$$\hat{\mathbf{a}} = \frac{1}{m}\mathbf{F} - g\hat{\mathbf{z}}^W, \quad \hat{\mathbf{g}} = \boldsymbol{\omega} \quad (20)$$

where \mathbf{F} is the net force in the body frame and g is the acceleration due to gravity, which acts in the $\hat{\mathbf{z}}^W$ direction. Finally, the encoders are used to measure the speeds of the motors and are described by

$$\hat{\mathbf{e}} = \boldsymbol{\Omega}, \quad \text{where } \boldsymbol{\Omega} = [\Omega_1, \Omega_2, \Omega_3, \Omega_4]^T \quad (21)$$

where Ω_j is the rotational speed of motor j . Note that sensor biases and noise have been omitted in this case for the purposes of clarity. The system output is then generally defined as

$$\mathbf{y} = [\hat{\mathbf{o}}_{\text{pos}}^T, \hat{\mathbf{o}}_{\text{att}}^T, \hat{\mathbf{a}}^T, \hat{\mathbf{g}}^T, \hat{\mathbf{e}}^T]^T \quad (22)$$

4.5 Guidance and Control

The rover is controlled by a state feedback controller and waypoint-based guidance module. For the purposes of brevity, the guidance and control algorithms are not detailed in this paper; however, it must be noted that a smooth input trajectory aids the InvSim process and the accuracy of its input estimation. More details on the controller are provided in [3].

5 SIMULATION TESTING

The use of InvSim to aid residual generation is tested in simulation. The model described in Section 4 informs both the system \mathbf{G} and the model of the system behaviour $\hat{\mathbf{G}}$, thus satisfying the condition $\mathbf{G} = \hat{\mathbf{G}}$. The two processes differ in that the system is subject to additive faults in the locations shown in Figure 6, while the model is not. As Figure 6 illustrates, the outputs of the system provide observations of both the rigid body dynamics and motor subsystems, through the IMU/Optitrack and encoders, respectively. As stated in Section 3, the InvSim algorithm operates most effectively when the number of inputs and outputs are equal. As $\mathbf{y} \in \mathbb{R}^{16}$ and $\mathbf{u} \in \mathbb{R}^2$, it is necessary to choose a subset $\mathbf{y}_t \in \mathbb{R}^2 \subset \mathbf{y}$ of the output in driving the InvSim algorithm. This subset depends on the location of the fault and the outputs which it excites.

Three test cases are considered, describing three distinct faults: a fault at the output of the system \mathbf{f}_y , a fault at the input to the system \mathbf{f}_u and a fault between the motor and rigid body subsystems \mathbf{f}_p . The first case is intended to demonstrate a simple example of existing residual generation techniques.

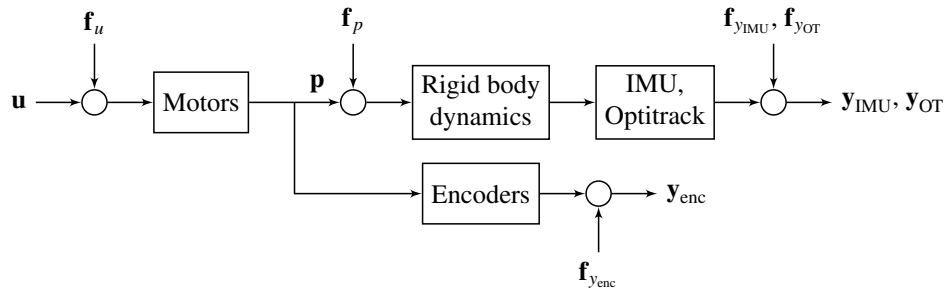


Figure 6: Generalised structure of the rover system.

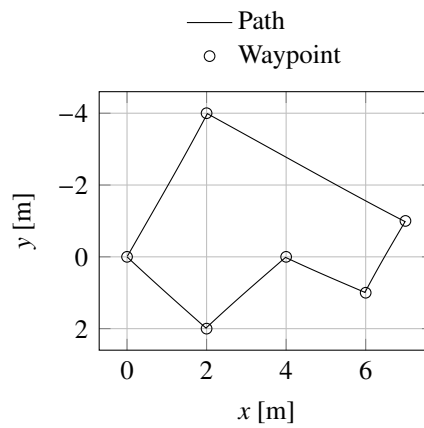


Figure 7: Path of rover through waypoints.

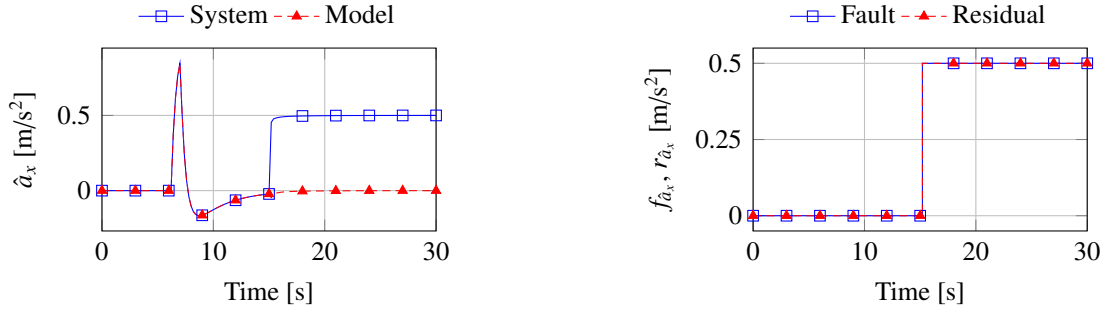
The latter two cases illustrate the use of InvSim to generate residuals elsewhere in the system, by providing additional estimates of variables from system measurements other than the two available inputs. In each case, the rover is instructed to follow a series of waypoints. These waypoints, and the fault-free rover's path through them, are shown in Figure 7. Motion is limited to the horizontal plane.

5.1 Detection of Output Faults

Consider an additive output fault in the accelerometer measurement in the x -axis, $f_{\hat{a}_x}$. The output fault is then

$$\mathbf{f}_y = \begin{bmatrix} \mathbf{0}_{6 \times 1} \\ f_{\hat{a}_x} \\ \mathbf{0}_{9 \times 1} \end{bmatrix}, \quad f_{\hat{a}_x} = \begin{cases} 0.5 \text{ m s}^{-2}, & t > 15 \text{ s} \\ 0 \text{ m s}^{-2}, & t \leq 15 \text{ s} \end{cases} \quad (23)$$

Equation (3) states that, for identical system and model, the output residual should be equivalent to an output fault in the same channel. This is confirmed by Figure 8. Figure 8a compares the true system output, clearly demonstrating a discontinuous change at $t = 15 \text{ s}$, with the modelled output. Figure 8b shows the residual in the accelerometer x -axis measurement, $r_{\hat{a}_x}$, matching the fault profile exactly. The remaining output residuals, omitted here, show no change when the fault occurs. The fault is thus easily isolated to the accelerometer measurement in x and its magnitude and shape determined.



(a) Comparison of system and estimated outputs.

(b) Comparison of output fault and residual.

Figure 8: Output signal and residual for a fault in the accelerometer measurement in the x -axis.

5.2 Detection of Input Faults

Consider an additive input fault in the voltage supply to the left-hand motors, f_{v_l} . The input fault is then

$$\mathbf{f}_u = \begin{bmatrix} f_{v_l} \\ 0 \end{bmatrix}, \quad f_{v_l} = \begin{cases} 1 \text{ V}, & t > 15 \text{ s} \\ 0 \text{ V}, & t \leq 15 \text{ s} \end{cases} \quad (24)$$

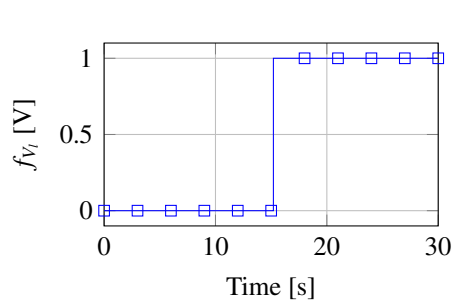
Output residual generation yields the results shown in Figure 9. Note that the Optitrack residuals have been omitted, while the remaining accelerometer and gyroscope residuals are zero due to the constrained motion of the rover. These residuals demonstrate a clear variance between system and model at $t = 15$ s, but do not otherwise provide useful information on the magnitude, shape or location of the fault.

Consider instead input residual generation. The InvSim algorithm is driven by a subset of the output \mathbf{y}_t , which provides sufficient observability of the fault location. Two different subsets are considered: the first at the output of the rigid body subsystem $\mathbf{y}_{t,\text{IMU}} = [\hat{a}_x, \hat{g}_z]^T$, describing the forward acceleration and yaw rate; the second at the output of the motor subsystem, $\mathbf{y}_{t,\text{enc}} = [\hat{e}_1, \hat{e}_4]^T$, describing the motorspeeds of the left and right motors. Equation (6) predicts that, for identical system and model, the input residual should be equivalent to an input fault in the same channel. This is confirmed by Figure 10. Figure 10a illustrates the divergence of the measured input signal from both InvSim estimates at the time of the fault occurring. Figure 10b validates the predicted result, with only slight perturbations exhibited in the two solutions for the residual r_{v_l} . In both cases, the residual clearly indicates the time, magnitude and shape of the fault, allowing the rover's recovery system to isolate and attenuate it.

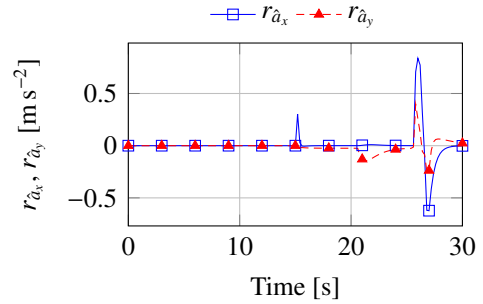
5.3 Detection of Faults Within Process

The process variable vector is defined to comprise the torques of each motor, that is

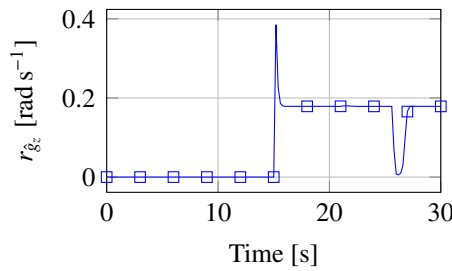
$$\mathbf{p} = [\tau_1, \tau_2, \tau_3, \tau_4]^T \quad (25)$$



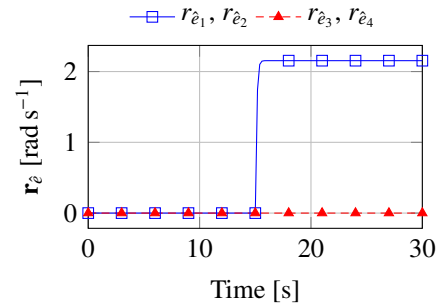
(a) Input fault in left-hand voltage, f_{V_l} .



(b) Accelerometer residuals in x - and y -axes, $r_{\hat{a}_x}$ and $r_{\hat{a}_y}$, respectively.

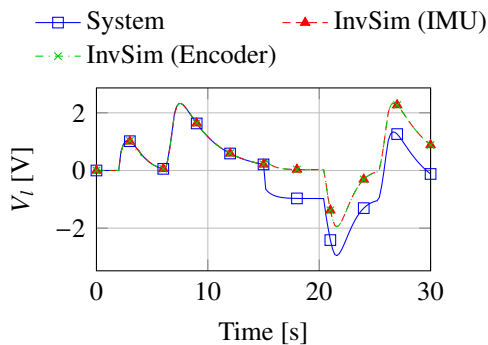


(c) Gyroscope yaw measurement residual, $r_{\hat{g}_z}$.

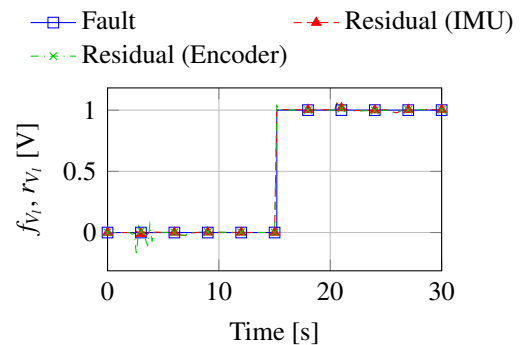


(d) Motor encoder residuals. Motors are numbered clockwise from the rear-left.

Figure 9: Input fault shape and output residuals for a fault in the voltage supply to the left-hand motors.

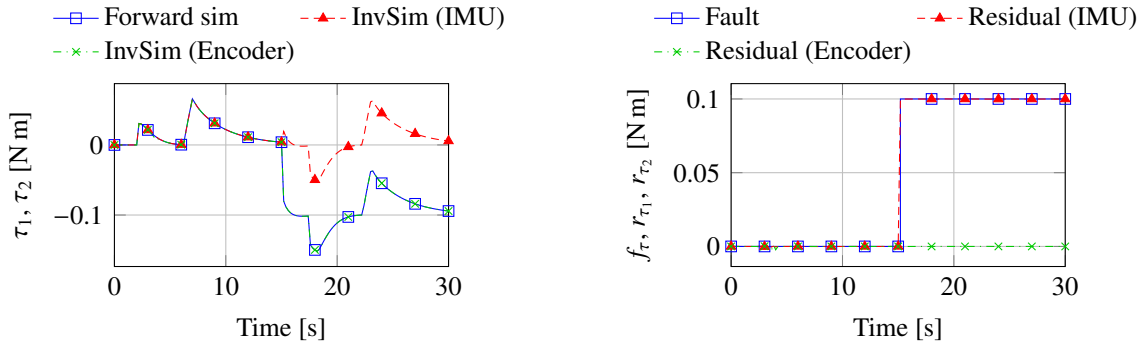


(a) Comparison of system and estimated inputs.



(b) Comparison of input fault and residual.

Figure 10: Input signal and residual for a fault in the voltage supply to the left-hand motors.



(a) Comparison of system and estimated variables.

(b) Comparison of process fault and residual.

Figure 11: Process variable and residual for a fault in the motor torques on the left-hand side of the rover.

Consider then identical additive input faults in the torques of the motors on the left-hand side of the rover, where the fault magnitude is r_τ . The process fault is then

$$\mathbf{f}_p = \begin{bmatrix} f_\tau \\ f_\tau \\ 0 \\ 0 \end{bmatrix}, \quad f_\tau = \begin{cases} 1 \text{ N m}, & t > 15 \text{ s} \\ 0 \text{ N m}, & t \leq 15 \text{ s} \end{cases} \quad (26)$$

Figure 6 highlights the location of \mathbf{p} as being between the motor and rigid body subsystems of the rover. A process (or equation error) residual may be generated to detect and locate the fault, using conventional and inverse simulations operating in tandem. The forward simulation is again driven by the two system inputs, while two separate InvSim algorithms are employed, driven by the output subsets $\mathbf{y}_{t,\text{IMU}}$ and $\mathbf{y}_{t,\text{enc}}$, respectively.

Equation (8) predicts that, for identical system and model and an invertible rigid body subsystem, the process residual between these subsystems will be equivalent to a fault at this location. This is confirmed in Figure 11. Figure 11a illustrates the divergence of the forward simulation estimate and the IMU-driven estimate at the time of the fault occurring. Figure 11b presents the residuals in the left-hand motor torques, τ_1 and τ_2 . The IMU-driven residual validates the predicted result, tracking the fault profile closely.

Conversely, the encoder-driven estimate does not diverge from the forward simulation estimate, as is the case in Figure 10a. As a consequence, the encoder-driven residual is negligible both before and after the fault. This may be attributed to the fact that the location of the fault \mathbf{f}_p is after the motor subsystem. The fault does not, therefore, affect the motor dynamics, which are observed by the encoders through the motorspeeds. It is clear from this result that the output subset must be chosen carefully, such that it is excited by the occurrence of the fault.

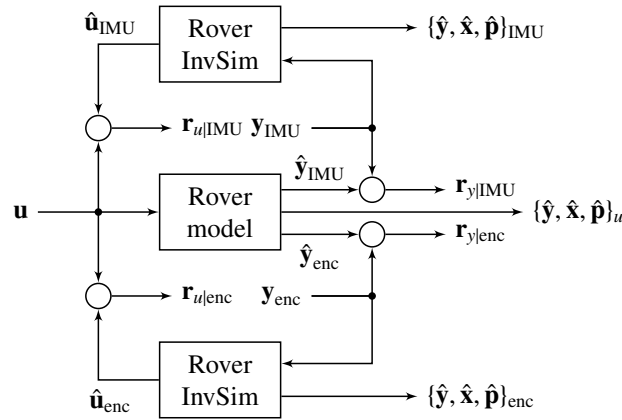


Figure 12: General architecture of simultaneous forward and inverse model residual generation.

6 PROPOSED ARCHITECTURE

Section 5 demonstrates the use of InvSim in providing useful residuals at specific locations in the system. However, the outputs which drive the InvSim algorithm must be matched dynamically to the fault and must therefore be chosen carefully. A comprehensive InvSim-based residual generation structure would thus comprise multiple parallel InvSim modules, driven by a multitude of outputs, augmenting the capability to detect and isolate the fault.

A proposed residual generation architecture for the rover is presented in Figure 12. Here, a forward model is combined with two inverted models: one driven by IMU outputs and the other driven by encoder outputs. In employing the encoder outputs to drive the inverse system, it is sufficient to consider only the subsystems which map the input to the driving output: here, the motor and encoder subsystems. However, there is a benefit to including the full system model: in the process of obtaining the input, the state, process variables and remaining outputs are also obtained. These may then be compared to corresponding results from the forward model or alternative InvSim module. Figure 12 highlights this approach: the general system variables $\{\hat{\mathbf{u}}, \hat{\mathbf{y}}, \hat{\mathbf{x}}, \hat{\mathbf{p}}\}$ for a given estimation method may be compared to those of any other method or the system measurements, providing a suite of residuals to work with. Naturally, a greater number of outputs corresponds to additional InvSim modules and therefore a greater variety of residuals.

A combination of forward simulation and multiple InvSims ultimately provides greater flexibility in the generation of features and subsequent production of analytical symptoms for use in fault diagnosis and recovery. While forward simulation is faster in execution than InvSim, it is driven only by the inputs and thus provides a single set of estimates during a given time period. In contrast, by exciting an inverted model from different output points, a multitude of estimates of a given variable are produced for a given time period. This is evident in the results shown in Figure 11. This approach is particularly beneficial when multiple faults occur in the system, by using measurements taken between faults. It may also be used to narrow thresholds in fault alarms, such as when one set of outputs can be trusted over another.

7 CONCLUSIONS

The results of Section 5 highlight the benefit of Inverse Simulation in providing greater flexibility in feature generation and a richer set of data from which to produce analytical symptoms for FDIR. The key advantage illustrated by the results presented in this paper is the clarity of the input and process residuals for specific faults in comparison to the output residuals for the same faults. Use of InvSim thus enables simpler detection and isolation of faults, while conventional forward simulation would require greater analysis of the generated residuals, including methods such as logical residual tables or adaptive thresholds.

Of note is that the type and location of the faults considered in this paper are deliberately simplistic. This facilitates initial feasibility testing of InvSim for use in residual generation. In cases where a fault occurs at a distinct division between subsystems, as illustrated in Figure 2, its detection and isolation are trivial. Where a fault occurs in a feedback loop or affects the system non-linearly, targeted residual generation becomes less trivial and requires direct coupling of forward and inverse models. Nevertheless, the ability to generate residuals at a variety of locations within the system can simplify the detection of such faults, even if the fault itself cannot be directly observed. This process, popularly known as equation error residual generation, is facilitated by InvSim. The *process residual* is analogous to equation error residual and used here to highlight its relationship to targeting specific process faults in the presented results.

InvSim may be used in standalone to produce estimates and residuals of the system inputs. It may also be used in combination with forward simulation, or, as highlighted in Section 6, with additional InvSim modules, to produce estimates of other system variables. Such variables include states, process variables and even other outputs. Section 6 discusses the use of different output measurements to excite the inverted model from different points. The IMU and encoder observe different parts of the system and are consequently affected by specific faults in different ways. Thus, the two InvSim modules presented in Figure 12 are as distinct from each other in the estimates they can produce as they are from the forward simulation. InvSim ultimately goes beyond providing a single estimation method to augment forward simulation, but provides a full suite of modules and corresponding estimates to bolster existing residual generation methods.

ACKNOWLEDGMENTS

The work in this paper was supported by a UK Space Agency CREST 3 award.

REFERENCES

- [1] E. Lakdawalla, "Brief Yutu update: Slightly more detail on what's keeping rover from roving — The Planetary Society," 2014. [Online]. Available: <http://www.planetary.org/blogs/emily-lakdawalla/2014/03030956-brief-yutu-update.html>
- [2] R. Isermann, *Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance*. Berlin: Springer-Verlag, 2006.

- [3] M. L. Ireland, K. J. Worrall, R. Mackenzie, T. Flessa, E. W. McGookin, and D. G. Thomson, "A Comparison of Inverse Simulation-Based Fault Detection in a Simple Robotic Rover with a Traditional Model-Based Method," in *19th International Conference on Autonomous Robots and Agents (ICARA 2017)*. Madrid: ICARA, March 2017.
- [4] K. Ferguson, "Towards a Better Understanding of the Flight Mechanics of Compound Helicopter Configurations," PhD, University of Glasgow, November 2015.
- [5] R. Hess, C. Gao, and S. Wang, "A generalized technique for inverse simulation applied to aircraft manoeuvres," *Journal of Guidance, Control and Dynamics*, vol. 14, no. 5, pp. 920–926, 1991.
- [6] D. Thomson and R. Bradley, "Inverse simulation as a tool for flight dynamics research – Principles and applications," *Progress in Aerospace Sciences*, vol. 42, no. 3, pp. 174–210, May 2006.
- [7] D. J. Murray-Smith, "The inverse simulation approach: a focused review of methods and applications," *Mathematics and Computers in Simulation*, vol. 53, no. 4-6, pp. 239–247, October 2000.
- [8] —, "Inverse simulation and analysis of underwater vehicle dynamics using feedback principles," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 20, no. 1, pp. 45–65, 2014.
- [9] D. J. Murray-Smith and E. W. McGookin, "A case study involving continuous system methods of inverse simulation for an unmanned aerial vehicle application," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 229, no. 14, pp. 2700–2717, 2015.
- [10] K. J. Worrall, D. G. Thomson, and E. W. McGookin, "Application of Inverse Simulation to a Wheeled Mobile Robot," in *6th International Conference on Automation, Robotics and Applications (ICARA 2015)*, Queenstown, February 2015.
- [11] K. J. Worrall, D. G. Thomson, E. W. McGookin, and T. Flessa, "Autonomous Planetary Rover Control using Inverse Simulation," in *13th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2015)*. Noordwijk: ESA/ESTEC, May 2015.
- [12] S. Rutherford and D. G. Thomson, "Improved methodology for inverse simulation," *Aeronautical Journal*, vol. 100, no. 993, pp. 79–85, 1996.
- [13] T. Flessa, E. W. McGookin, and D. G. Thomson, "Numerical stability of inverse simulation algorithms applied to planetary rover navigation," in *24th Mediterranean Conference on Control and Automation, MED 2016*. Athens: IEEE, June 2016, pp. 901–906.
- [14] T. Flessa, E. McGookin, D. Thomson, and K. Worrall, "Numerical Efficiency of Inverse Simulation Methods Applied to a Wheeled Rover," in *Proceedings of the 9th EUROSIM Congress on Modelling and Simulation*. Oulu: EUROSIM, September 2016.
- [15] K. J. Worrall, "Guidance and Search Algorithms for Mobile Robots: Application and Analysis Within the Context of Urban Search and Rescue," PhD, University of Glasgow, September 2008.